浙江大学

本科实验报告

课程名称：计算机组成
姓名：曾一欣

学院：竺可桢学院

班级：求是科学班（计算机科学与技术）

专业：计算机科学与技术

学号：3180105144
指导老师：姜晓红
2020.5.6

浙江大学实验报告

课程名称： 计算机组成　　实验类型：综合

实验项目名称：Lab3 Single Cycle CPU

学生姓名：曾一欣　　专业：计算机科学与技术　　学号： 3180105144

同组学生姓名： None　　指导老师： 姜晓红

实验地点：无　　实验日期： 2020 年 5 月 6 日

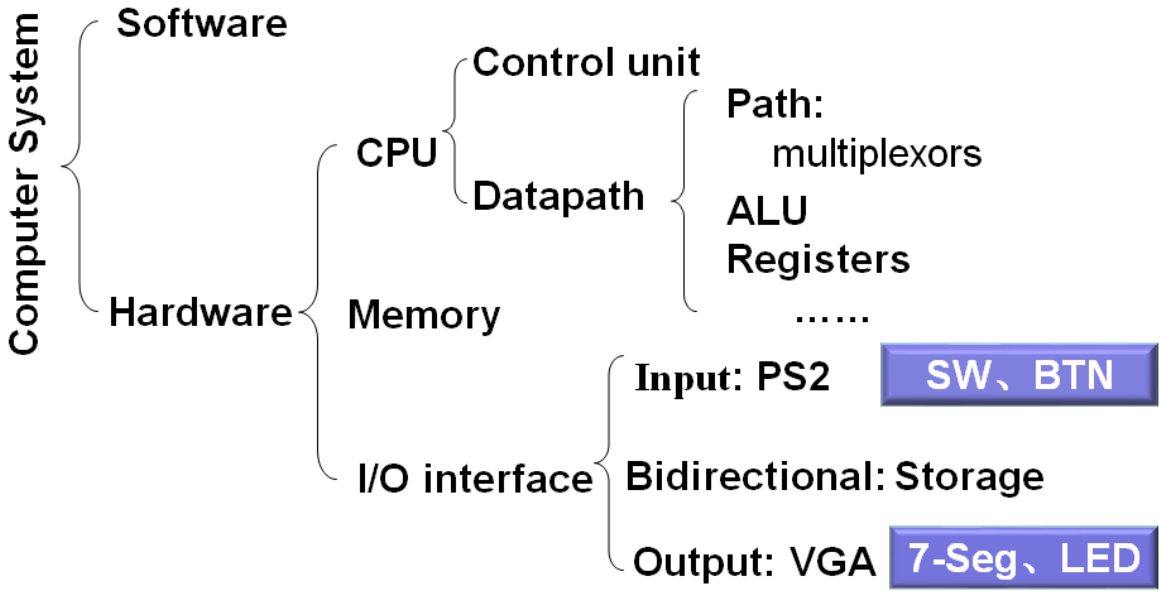
note: In this experiment I designed the SCPU by myself instead of follow the instructions in the slides. So the report might be different from lab PPTs because my steps to construct the SCPU is different.

一、 实验目的和要求

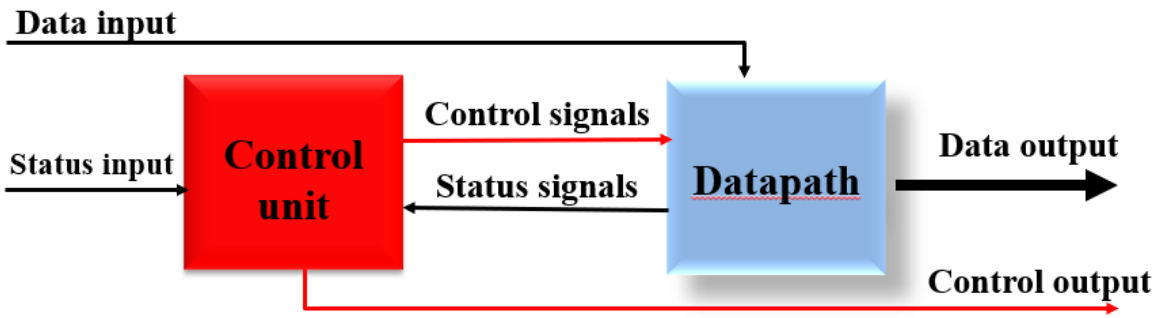
1. 构建数据通路
2. 构建控制器
3. 设计数据通路的核心部件
4. 连接完成SCPU

二、 实验内容和原理

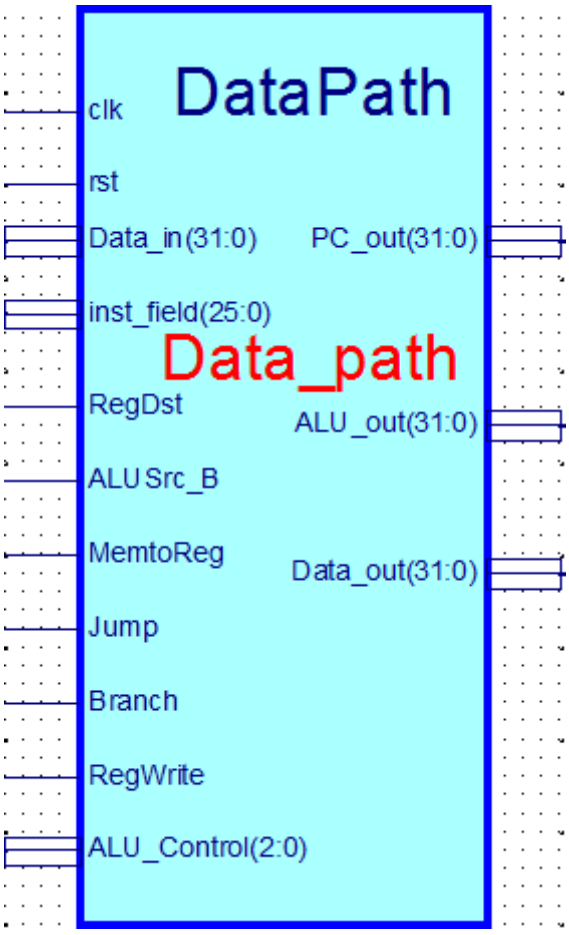
Decomposability of computer systems



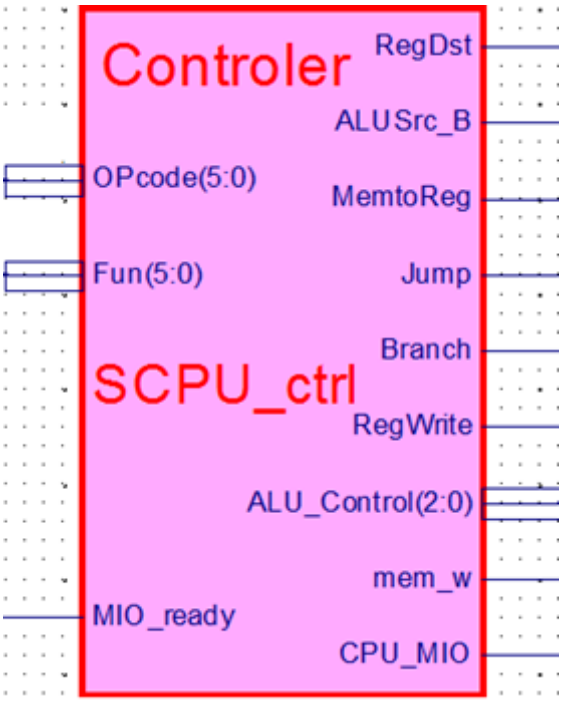
Digital circuit



DataPath

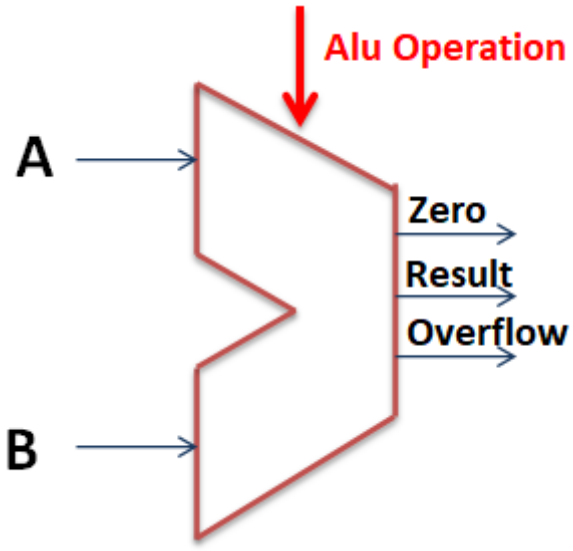


Controller

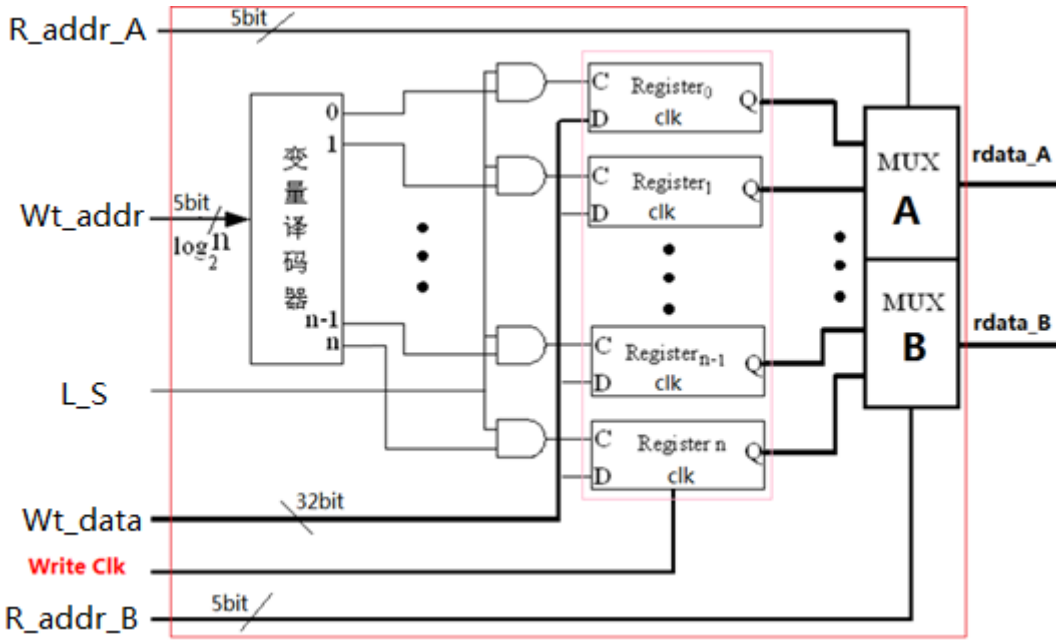


ALU

ALU Control Lines	Function	note
000	And	兼容
001	Or	兼容
010	Add	兼容
110	Sub	兼容
111	Set on less than	
100	nor	扩展
101	srl	扩展
011	xor	扩展

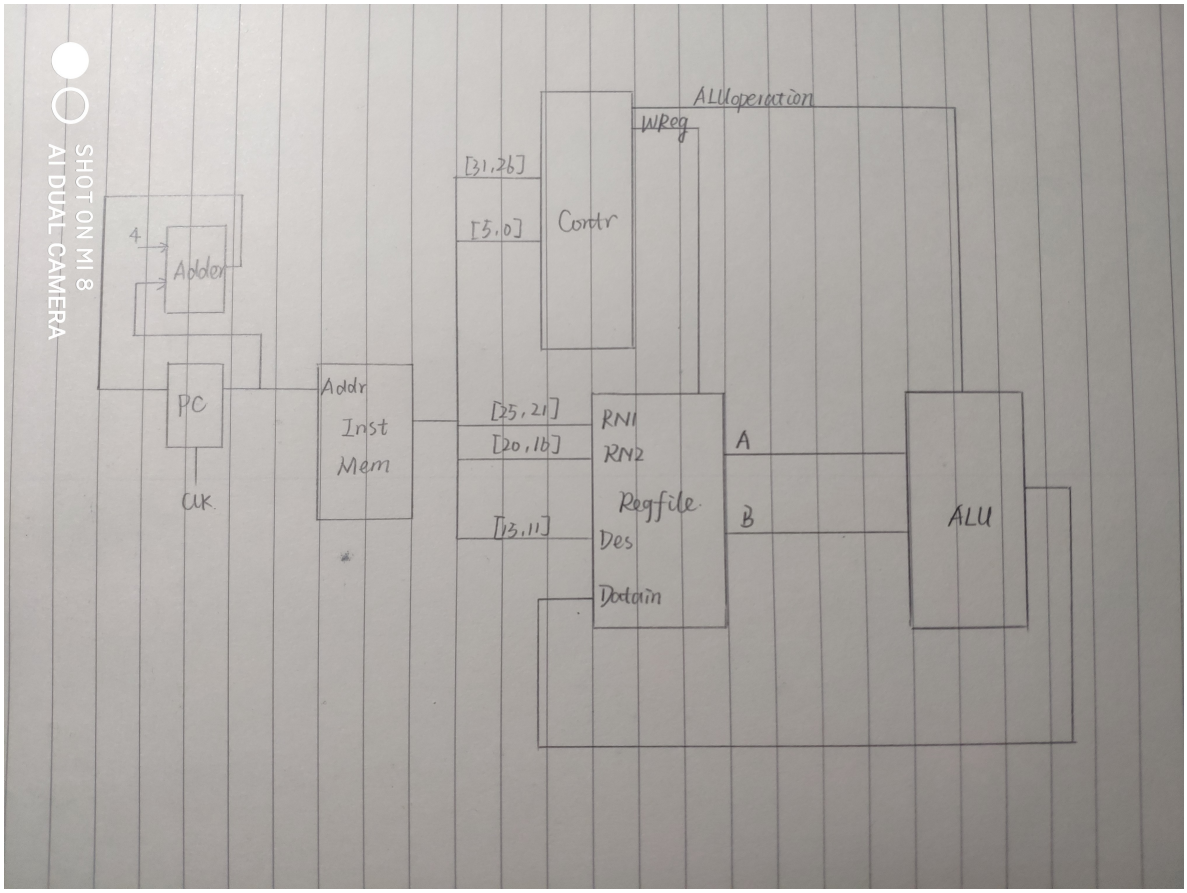


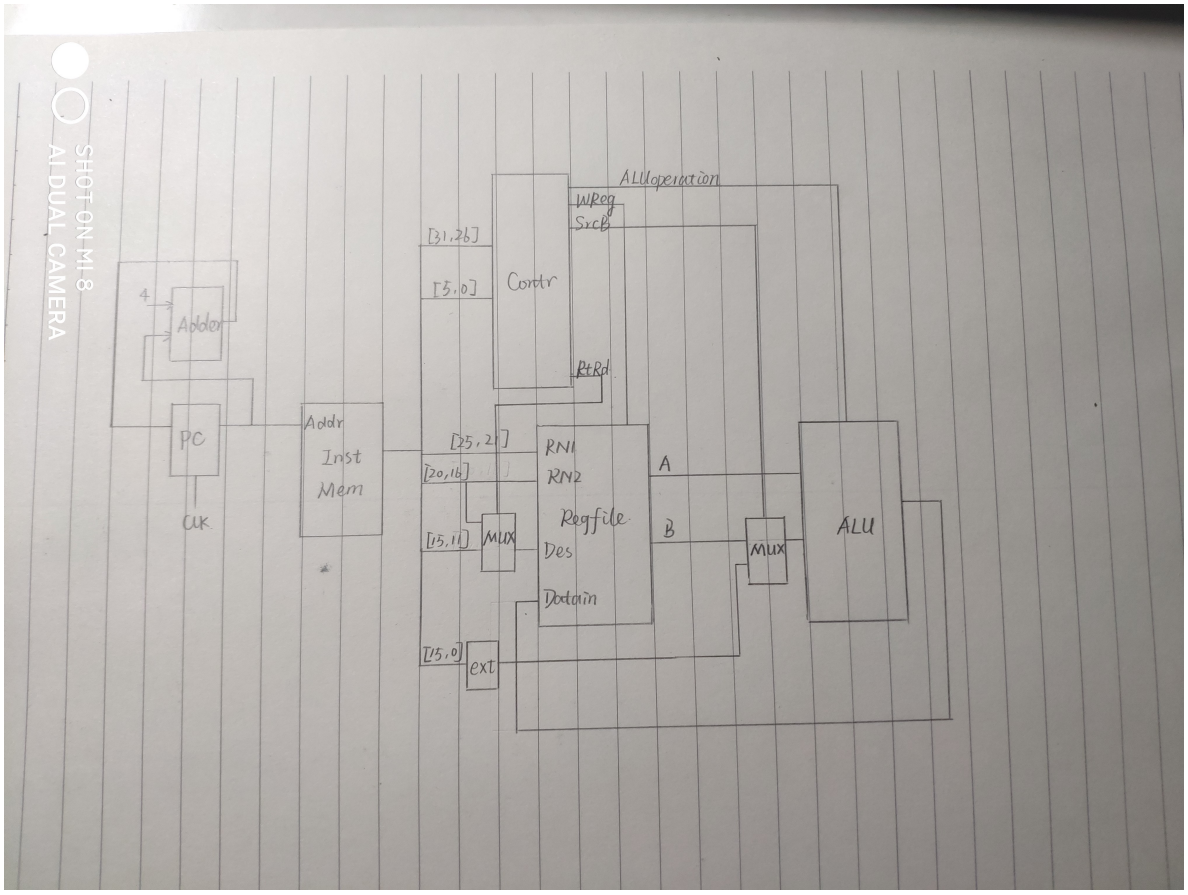
Register file

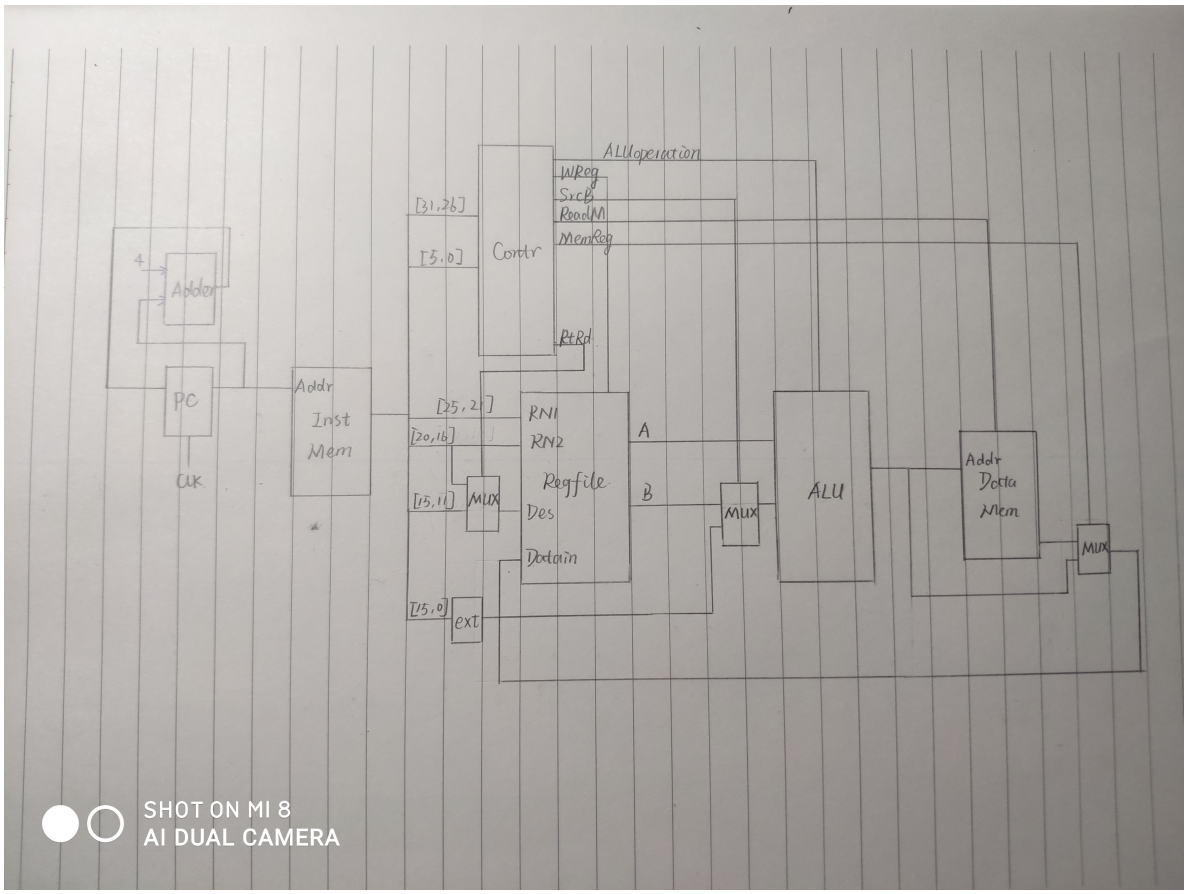


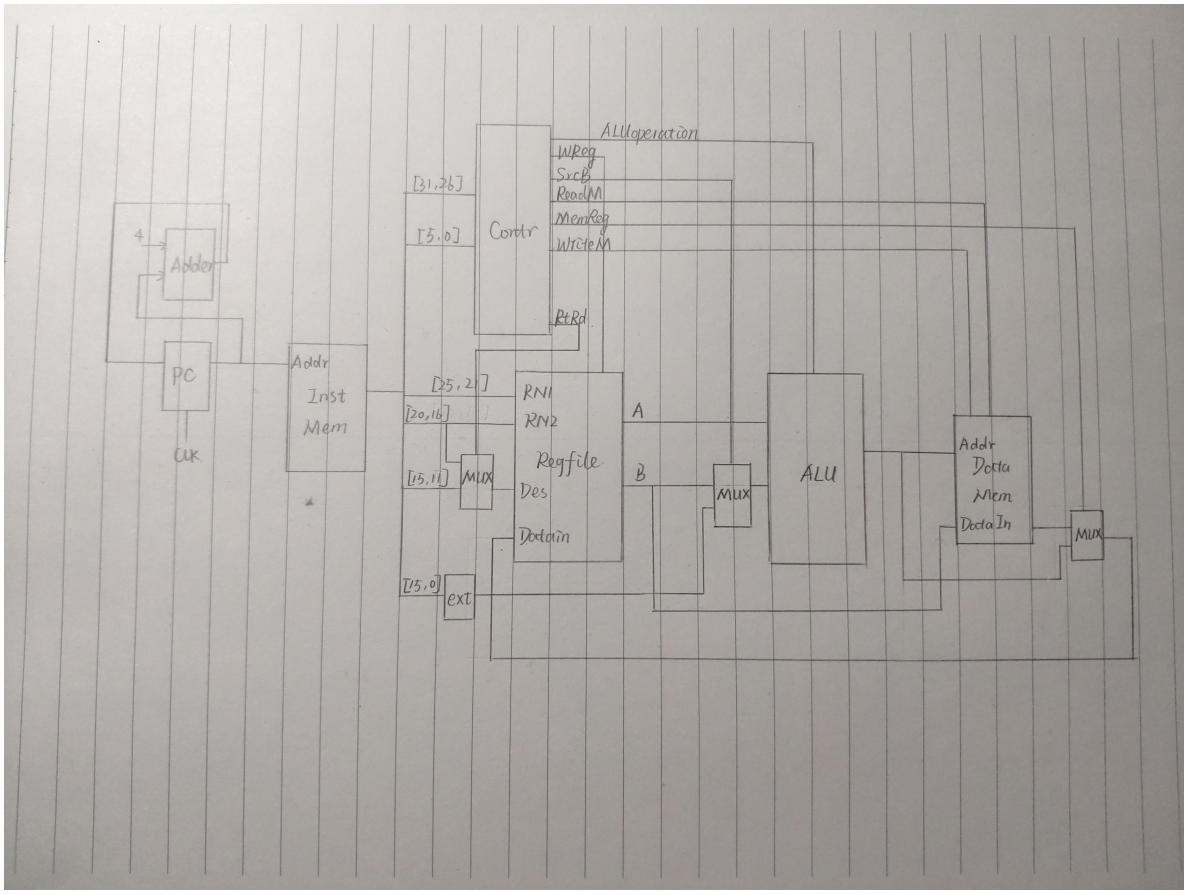
三、 实验过程和数据记录及结果分析

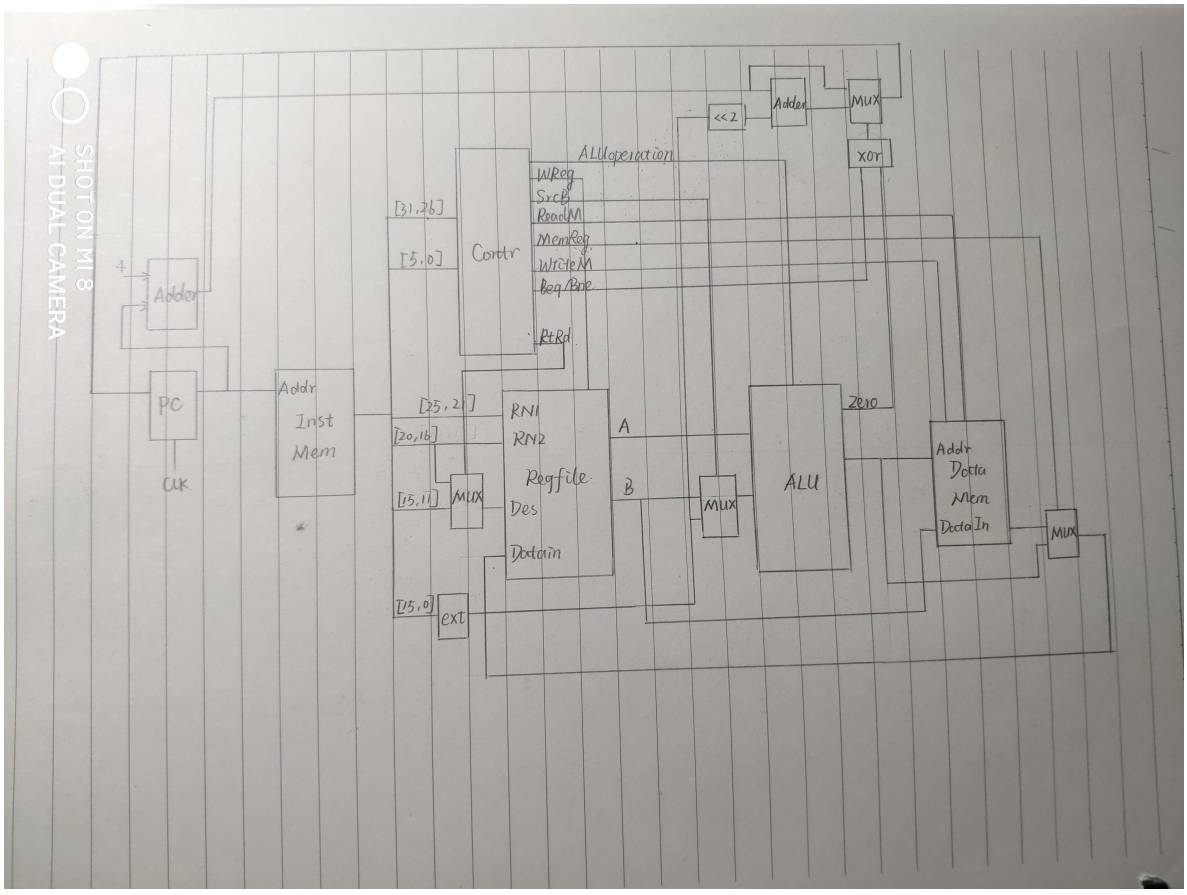
Construct the circuit

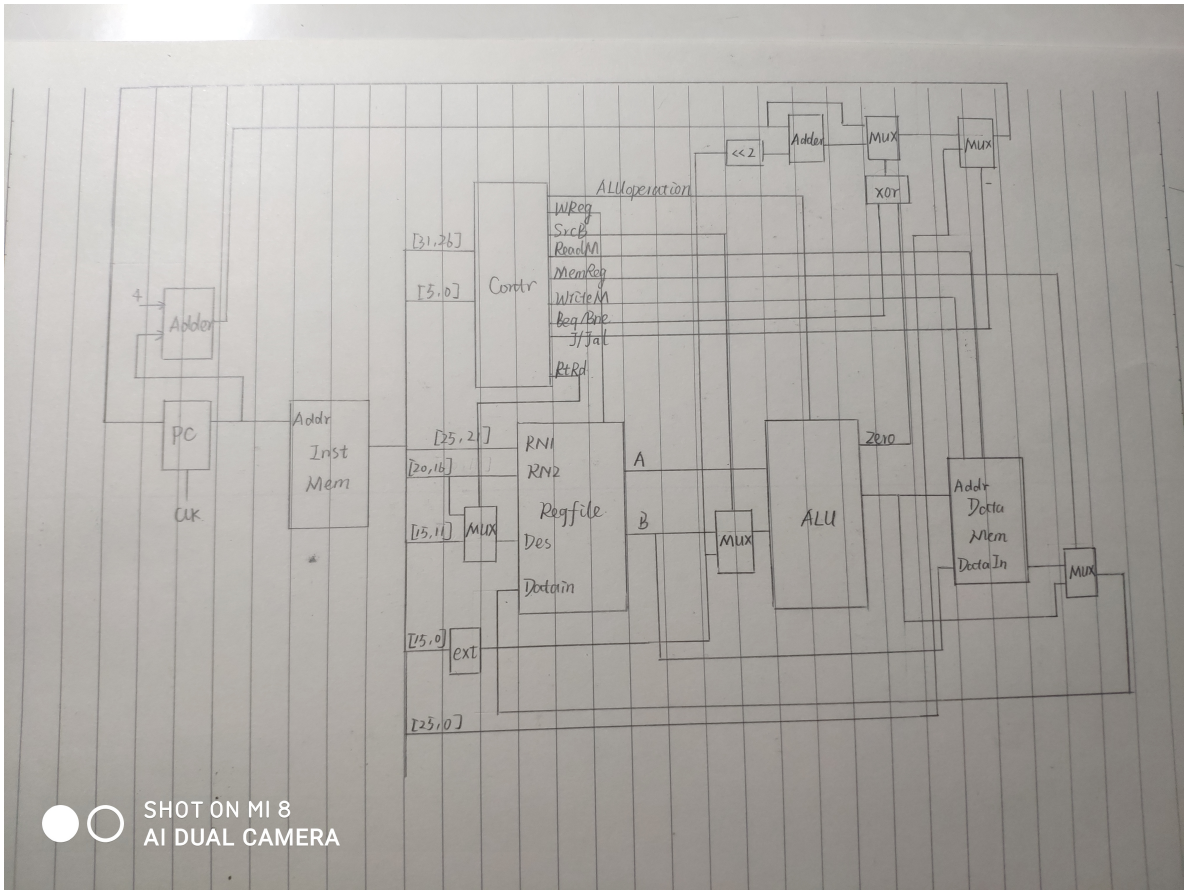


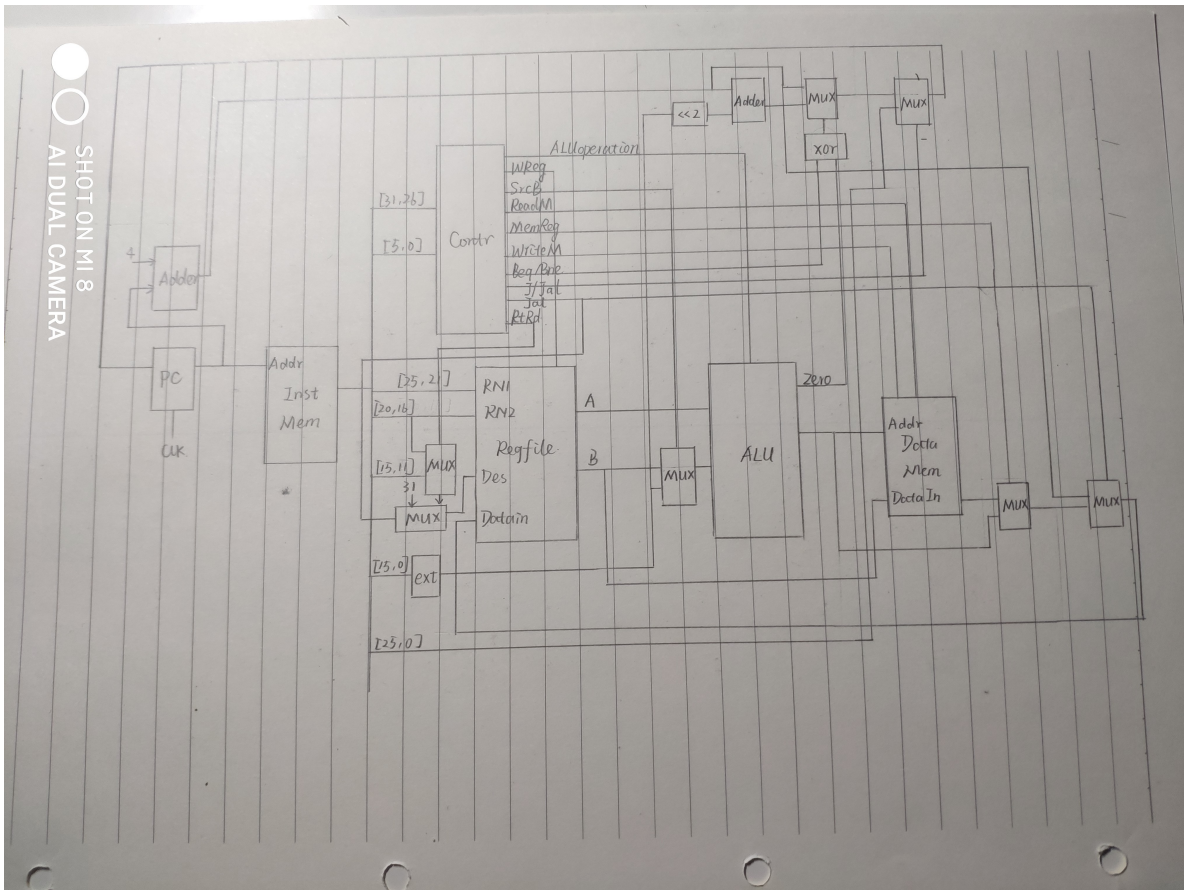


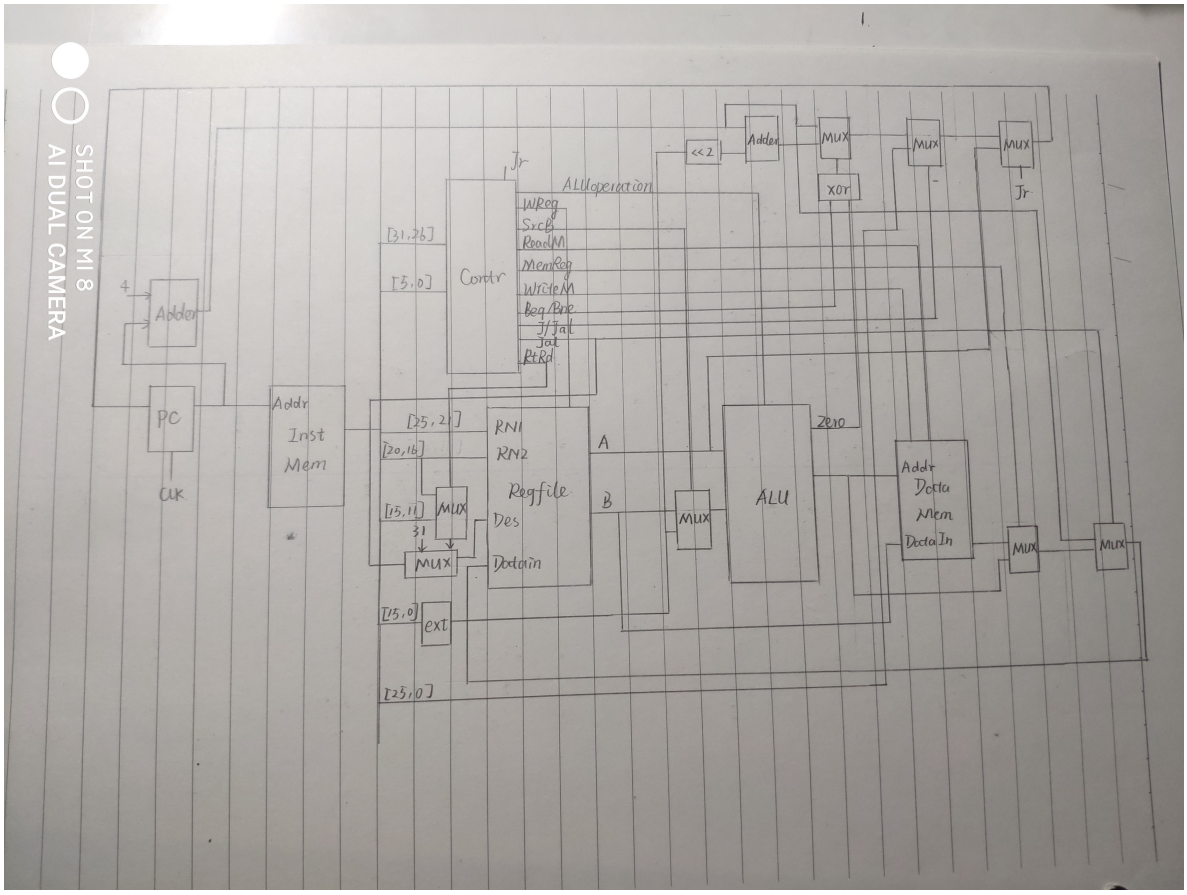


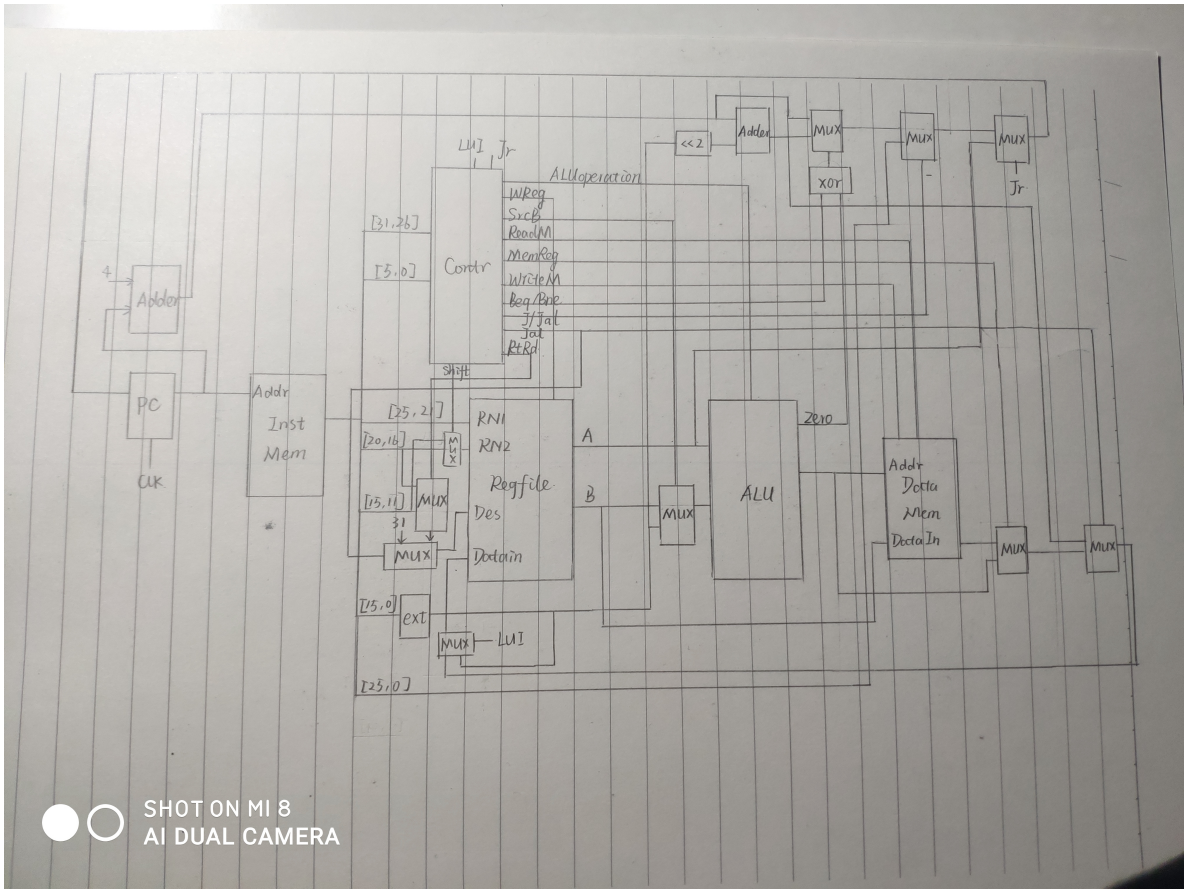

ALUoperation
WReg
SrcB
RdadM
MemReg
WriteM
RtRd
[31,26]
[5,0]
Contr
4
Adder
PC
clk
Addr
Inst
Mem
[25,21]
[20,16]
[15,11]
RN1
RN2
Regfile
Des
Datain
MUX
A
B
ALU
Addr
Data
Mem
DataIn
[15,0]
ext


SHOT ON MI 8
AI DUAL CAMERA
<<2
Adder
MUX
XOR
ALUoperation
WReg
SrcB
RdadM
MemReg
WriteM
RegDst
RtRd
[31,26]
[5,0]
Contr
4
Adder
PC
clk
Addr
Inst
Mem
[25,21]
[20,16]
[15,11]
RN1
RN2
Regfile
Des
Datain
MUX
A
B
ALU
zero
Addr
Data
Mem
DataIn
[15,0]
ext









Modules

ALU_SCPU

use the module ALU4b last semester

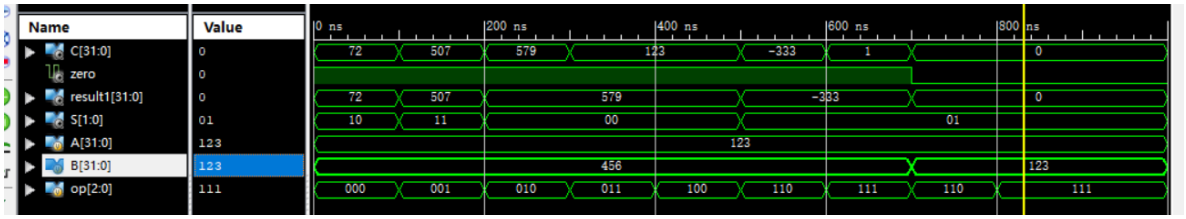
```
module ALU32b(
```

```
        input [31:0]A,
        input [31:0]B,
        input [1:0]S,
        output [31:0]C,
        output Co
    );

wire C0;
ALU4b m1(.A(A[3:0]),.B(B[3:0]),.S(S),.C(C[3:0]),.Ci(S[0]),.Co(C0));
wire C1;
ALU4b m2(.A(A[7:4]),.B(B[7:4]),.S(S),.C(C[7:4]),.Ci(C0),.Co(C1));
wire C2;
ALU4b m3(.A(A[11:8]),.B(B[11:8]),.S(S),.C(C[11:8]),.Ci(C1),.Co(C2));
wire C3;
ALU4b m4(.A(A[15:12]),.B(B[15:12]),.S(S),.C(C[15:12]),.Ci(C2),.Co(C3));
wire C4;
ALU4b m5(.A(A[19:16]),.B(B[19:16]),.S(S),.C(C[19:16]),.Ci(C3),.Co(C4));
wire C5;
ALU4b m6(.A(A[23:20]),.B(B[23:20]),.S(S),.C(C[23:20]),.Ci(C4),.Co(C5));
wire C6;
ALU4b m7(.A(A[27:24]),.B(B[27:24]),.S(S),.C(C[27:24]),.Ci(C5),.Co(C6));
ALU4b m8(.A(A[31:28]),.B(B[31:28]),.S(S),.C(C[31:28]),.Ci(C6),.Co(Co));

endmodule
```

```
module ALUSCPU(
input [31:0]A,
input [31:0]B,
input [2:0]op,  //operation
output [31:0]C,
output zero,
output [1:0]S,
output [31:0]result1
    );

wire [1:0]S=0;
assign S[0]=(op==6||op==1||op==7)?1:0;
assign S[1]=(op==0||op==1)?1:0;
wire co;
wire [31:0]result1; //ALU result
ALU32b m1(A,B,S,result1,co);

wire [31:0]result2;  //SLL result
wire [31:0]result3; //SRL result
wire [31:0]result4; //SLT result
assign result4=(result1[31]==1)?1:0;
SLL m2(A,B[10:6],result2);
SRL m3(A,B[10:6],result3);

wire [31:0]temp1;
wire [31:0]temp2;
assign temp1=(op==7)?result4:result1;
assign temp2=(op==4)?result3:result2;

assign C=(op==3||op==4)?temp2:temp1;
```

```
assign
zero=C[0]|C[1]|C[2]|C[3]|C[4]|C[5]|C[6]|C[7]|C[8]|C[9]|C[10]|C[11]|C[12]|C[13]|C
[14]|C[15]|C[16]|C[17]|C[18]|C[19]|C[20]|C[21]|C[22]|C[23]|C[24]|C[25]|C[26]|C[2
7]|C[28]|C[29]|C[30]|C[31];
endmodule
```

simulation:



Regfile

```
module Regfile(
input rst,
input clk,
input Wreg,
input [4:0]RN1,
input [4:0]RN2,
input [4:0]Des,
input [31:0]datain,
output [31:0]A,
output [31:0]B
    );
reg [31:0]a[1:31];
integer i;
assign A=(RN1==0)?0:a[RN1];
assign B=(RN2==0)?0:a[RN2];

always@(posedge rst)
begin
    for(i=1;i<32;i=i+1)
        a[i]<=0;
end

always@(negedge clk)
begin
    if(Des!=0&&Wreg==1)
        a[Des]<=datain;
end

endmodule
```

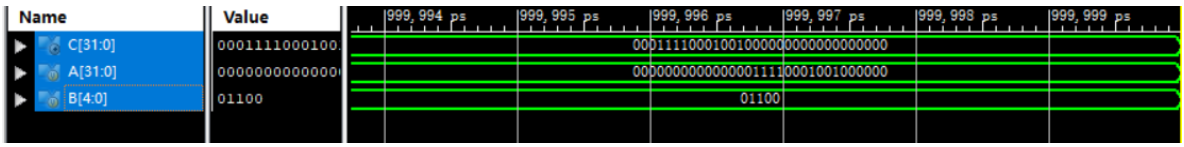
Controller

```
module FirstDecoder(
input [5:0]opcode,
input [5:0]func,
output [2:0]operation,
output LUI,
output Jr,
output Wreg,
output SrcB,
```

```
output ReadM,
output MemReg,
output WriteM,
output Beq,
output Bne,
output JJal,
output Jal,
output RtRd,
output shift
 );

wire ALU;
wire ALURI;
wire LW;
wire SW;
wire BEQ;
wire BNE;
wire J;

assign ALURI=(opcode[5:3]==1&&opcode[2:0]!=7)?1:0;
assign ALU=((opcode==0&&func!=8)||ALURI==1)?1:0;
assign LW=(opcode==6'b100011)?1:0;
assign SW=(opcode==6'b101011)?1:0;
assign BEQ=(opcode==6'b000100)?1:0;
assign BNE=(opcode==6'b000101)?1:0;
assign J=(opcode==2)?1:0;
assign Jal=(opcode==3)?1:0;
assign LUI=(opcode==15)?1:0;
assign Jr=(opcode==0&&func==8)?1:0;

assign operation[0]=((opcode==0&&(func==0||func==6'b101010||func==8))||
(opcode==13)||(opcode==10))?1:0;
assign operation[1]=((opcode==0&&
(func==32||func==34||opcode==0||func==6'b101010))||(opcode==8)||
(opcode==6'b100011)||(opcode==6'b101011)||(opcode==6'b000100)||
(opcode==6'b000101)||(opcode==6'b001010))?1:0;
assign operation[2]=((opcode==1&&(func==34||func==6'b101010||func==2))||
(opcode==6'b000100)||(opcode==6'b000101)||(opcode==6'b001010))?1:0;

assign shift=(opcode==0&&(func==0||func==2))?1:0;
assign Wreg=(ALU==1||LW==1||Jal==1||LUI==1||shift==1)?1:0;
assign SrcB=(ALURI==1||LW==1||SW==1)?1:0;
assign ReadM=LW;
assign MemReg=LW;
assign WriteM=SW;
assign Beq=BEQ;
assign Bne=BNE;
assign JJal=J&Jal;
assign RtRd=(ALURI||LW||LUI)?1:0;

endmodule
```

SLL

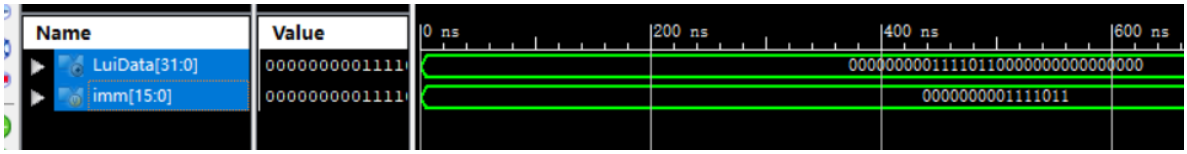
I try to calculate the 32 result of SLL from 0 to 31 and use a 32T1 MUX to output the asked one. I thought this method is not good and wish to learn a better way to do SLL from others.



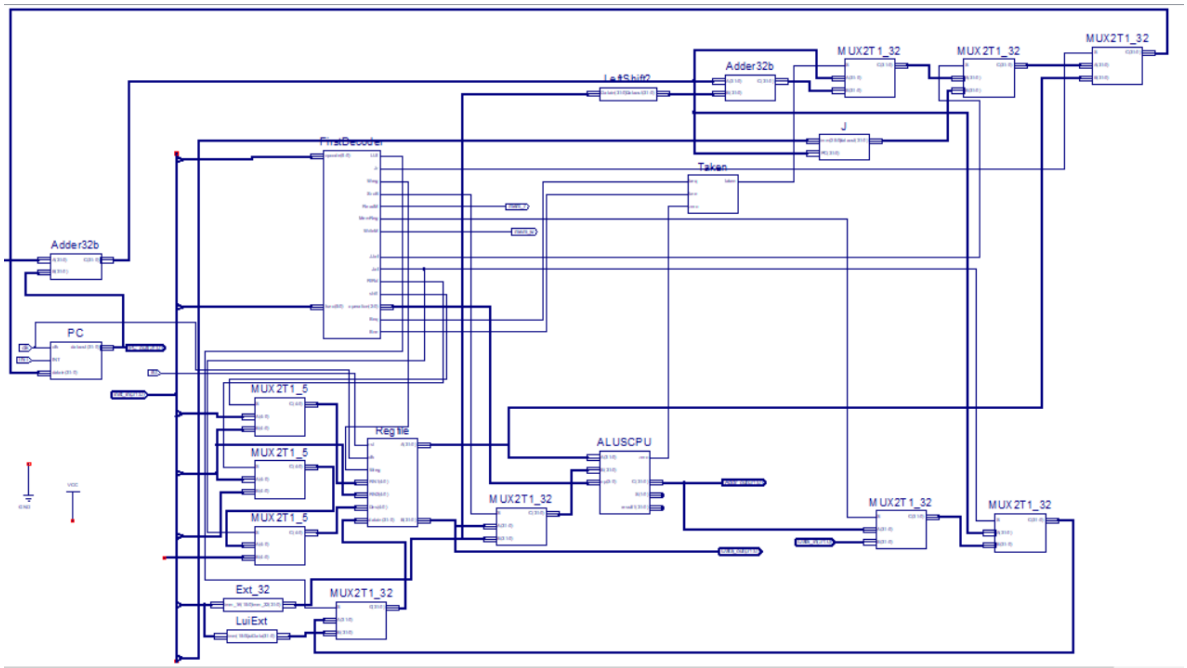
others

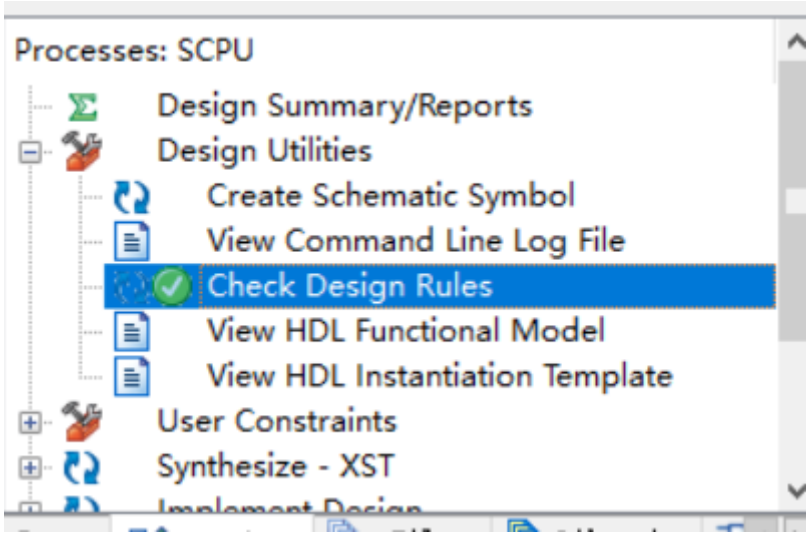
some other modules are simulated in the first experiment.

```
module LuiExt(
input [15:0]imm,
output [31:0]LuiData
    );
assign LuiData[31:16]=imm;
assign LuiData[15:0]=0;

endmodule
```



Wire





四、 讨论与心得

As the datapath and control module of single cycle CPU is really important and the basic idea on how to construct it is taught in the class, I try to do this experiment without the help of lab slides. It takes long time to think about the correct datapath and I hope I can make it run successfully.

There's two module that I can't find the right way to complete: SLL and SRL. I can't think of a easier way than calculate the result of SLL/SRL from 1 to 32 bits and use a MUX to get the result. But I thought this MUX methond is not correct. So I leave the two module blank there and hope I could solve these quesstion later with some new knowledge learned.

Finally, I find some stange phenomena during simulation. In the simulation of ALU_SCPU, if I output wire "result1" and "S" which is actually useless to outer modules, I can get a right simlation result; if I don't, I will get wrong result witch shows me red lines in four of the simulation input values. I can't figure out why this happens, all I can do is to keep the two useless output wire but make them unconnected in the top module.

update in 5.11

I complete SLL part by calculating the 32 result of SLL from 0 to 31 and then using a 32T1 MUX to output the asked one because the teacher mentioned that we are required to do this part in dingding. And I also do simulation for it and proved that it's correct. But I don't think it's the right way to do it, and I'm still waiting for a better one so I remained the SRL part empty.